El sistema de compresión JPEG. Un pequeño paseo por la transformada discreta de Fourier y la coseno

por

Julio Benítez López

Resumen. Este artículo revisa el sistema de compresión de imágenes JPEG. Este sistema se basa en la transformada discreta coseno, que a su vez, se basa en la transformada discreta de Fourier. Ambas transformadas se introducen de forma razonada y se explica el uso de la transformada discreta coseno al sistema de compresión de los archivos JPEG.

1. Introducción

Todos nosotros sabemos lo que es un archivo JPEG: cuando hacemos una fotografía con el móvil o con cualquier cámara digital, esta es guardada con esta extensión. JPEG es de hecho el formato estándar y el más popular de los usados en internet. ¿Por qué? No es nada evidente al ojo humano, pero el formato JPEG "sacrifica" información para hacer que los ficheros JPEG sean pequeños. Cada vez que hacemos una foto y la guardamos en este formato, algunos datos se pierden; pero esta pérdida de información es indistinguible al ojo humano. Este sistema se contrapone a los ficheros BMP (usados originalmente por Windows). BMP no permite ninguna compresión: se almacena toda la información píxel a píxel, por tanto son precisos al 100 %... pero los ficheros BMP son enormes. Una misma fotografía guardada con el formato JPEG permite ahorrar cerca de un 90 % de memoria en comparación con BMP. Por eso no se usa apenas el formato BMP.

Además hay otros formatos gráficos que no estudiaremos, como el PNG o el GIF, cada uno con sus ventajas e inconvenientes. Solo estudiaremos imágenes en blanco y negro, ya que las imágenenes en color necesitan de pocos ajustes adicionales más.

JPEG (del inglés *Joint Photographic Experts Group*) es el nombre de un comité de expertos que creó en los años 90 un estándar de compresión y codificación de imágenes. Su página web es `http://www.jpeg.org/index.html`

2. Transformada discreta de Fourier

La representación de una función como una serie de Fourier ha encontrado infinidad de aplicaciones tanto en el mundo de la técnica como en las matemáticas puras. Es bien conocido que bajo determinadas condiciones (que no detallaremos aquí,

pues se pueden encontrar en cualquier texto sobre series de Fourier), una función T-periódica, sea $h : \mathbb{R} \to \mathbb{R}$, se puede representar mediante

$$h(t) = \sum_{n \in \mathbb{Z}} a_n \mathrm{e}^{2\pi n i t/T}. \tag{1}$$

En las aplicaciones prácticas no se conoce el comportamiento completo de una señal T-periódica: se suele conocer el valor de la señal en una serie de valores concretos. Más concretamente, dada la señal $h : [0, T] \to \mathbb{R}$, lo que solo se suele conocer son los valores $h(0)$, $h(T/n)$, $h(2T/n)$, ..., $h((n-1)T/n)$.

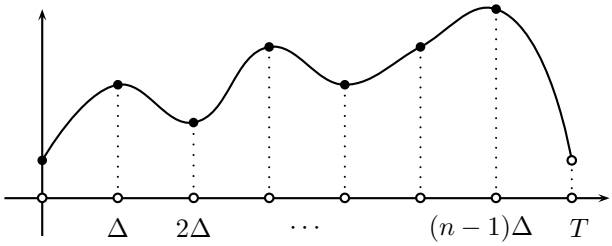

Figura 1: Discretización de una señal T-periódica en n muestras. Aquí, $\Delta = T/n$.

Un situación que surge y hay que tener en cuenta es el *aliasing*. Consiste en tener dos señales T-periódicas distintas, $h, g : [0, T] \to \mathbb{R}$, tales que $h(kT/n) = g(kT/n)$ para $k = 0, \ldots, n-1$. Estas dos señales son indistinguibles con esta discretización. Este fenómeno causa que las aspas de un ventilador parezcan a veces girar en el sentido inverso del que en realidad lo hacen, cuando se les filma o cuando son iluminadas por una fuente de luz parpadeante. Véanse la figuras 2 y 3. A veces, en las películas, las ruedas de los carros parecen girar en sentido contrario al que deberían.

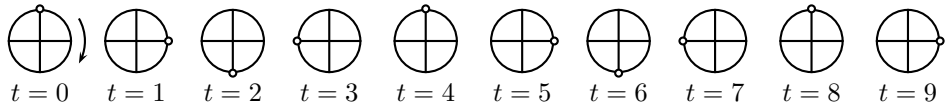

Figura 2: La imagen simboliza las aspas de un ventilador (que se mueve en sentido a favor de las agujas del reloj). Si solo vemos el ventilador en $t = 0, 4, 8, \ldots$, creeremos que está parado. Si lo vemos en $t = 0, 3, 6, 9, \ldots$, creeremos que las aspas van en sentido contrario a las agujas del reloj.

Una pregunta interesante es la siguiente: dada una señal, ¿cuál debe ser la frecuencia de muestreo para evitar el aliasing? Es decir, para recuperar de forma inequívoca la señal. La respuesta nos la proporciona el teorema de Nyquist-Shannon.

Teorema 1 (Teorema de Nyquist-Shannon). *Una señal sin frecuencias mayores que F Hz se puede representar de forma exacta especificando los valores de la señal en instantes de tiempo separados por $1/(2F)$ segundos.*

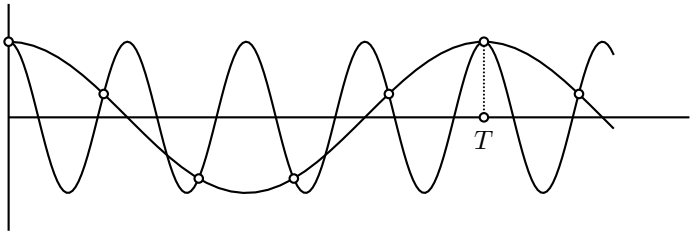

Figura 3: Fenómeno del *aliasing*: las dos señales son indistinguibles. El cerebro tiende a suponer que la imagen que percibe es la de menor frecuencia.

Se puede consultar la demostración en [1].

Un error extendido es creer que al aumentar la tasa de muestreo se mejora la calidad de la señal recuperada: una vez cumplido el criterio del teorema de Nyquist-Shannon, la reconstrucción es exacta (al menos desde el punto de vista matemático). Así, una señal que represente la voz humana no suele tener información relevante más allá de los 10 kHz, y de hecho en telefonía fija se toman solo los primeros 3'8 kHz. Con 2 kHz basta para que la voz sea comprensible, pero no para reconocer al hablante. Por tanto, si deseamos grabar una conversación telefónica, para no perder información, deberemos tomar del orden de 7600 muestras por segundo.

Nota: Desde ahora, los índices de los vectores y matrices empezarán desde 0. Los superíndices t y $*$ denotarán la traspuesta y conjugada traspuesta de una matriz, respectivamente. Los vectores de $\mathbb{R}^n$ o bien $\mathbb{C}^n$ se sobrentenderán que son columnas y si $\mathbf{v} \in \mathbb{R}^n$ o $\mathbf{v} \in \mathbb{C}^n$, entonces $\mathbf{v}[k]$ denotará la componente k-ésima de $\mathbf{v}$.

Una vez discretizada la señal $h(t)$ en los valores $t = 0, \frac{T}{n}, \frac{2T}{n}, \ldots, \frac{(n-1)T}{n}$ obtenemos el vector $\mathbf{h} = [\mathbf{h}[0], \cdots, \mathbf{h}[n-1]]^t \in \mathbb{R}^n$ y aplicamos (1):

$$\mathbf{h}[k] = h(kT/n) = \sum_{m\in\mathbb{Z}} a_m \mathrm{e}^{2\pi m\mathrm{i}k/n}, \qquad k = 0, 1, \ldots, n-1. \tag{2}$$

Ahora bien, si dividimos cualquier entero m entre n, obtenemos un cociente c y un resto r. Estos números cumplen $r \in \{0, 1, \ldots, n-1\}$ y $m = c \cdot n + r$. Por lo que

$$\exp\left(\frac{2\pi m\mathrm{i}k}{n}\right) = \exp\left(\frac{2\pi(cn+r)\mathrm{i}k}{n}\right) = \exp\left(\frac{2\pi r\mathrm{i}k}{n}\right),$$

ya que $\exp(2\pi\mathrm{i}q) = 1$ para cualquier entero q. Por tanto, el sumatorio infinito que aparece en (2) se convierte en un sumatorio finito:

$$\mathbf{h}[k] = \sum_{r=0}^{n-1} b_r \mathrm{e}^{2\pi r\mathrm{i}k/n}, \tag{3}$$

siendo $b_r = \sum_{c\in\mathbb{Z}} a_{c+rn}$. Resulta que esta última expresión nos es absolutamente irrelevante y la única que de verdad va a ser importante es la (3). Ahora el problema es, dados n valores $\mathbf{h}[0], \ldots, \mathbf{h}[n-1]$, ¿cómo encontrar los valores $b_0, \ldots, b_{n-1}$ para que se cumpla (3)?

En primer lugar, expresamos (3) en forma matricial: Si llamamos $\omega = \mathrm{e}^{2\pi \mathrm{i}/n}$, entonces (3) se reescribe como $\mathbf{h}[k] = b_0 + b_1\omega^k + b_2\omega^{2k} + \cdots + b_{n-1}\omega^{(n-1)k}$, luego

$$\begin{bmatrix} \mathbf{h}[0] \\ \mathbf{h}[1] \\ \mathbf{h}[2] \\ \vdots \\ \mathbf{h}[n-1] \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^{n-1} \\ 1 & \omega^2 & \omega^4 & \cdots & \omega^{2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{n-1} & \omega^{2(n-1)} & \cdots & \omega^{(n-1)^2} \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ \vdots \\ b_{n-1} \end{bmatrix}. \tag{4}$$

El vector $[b_0, b_1, \ldots, b_{n-1}]^t$ se llama *la transformada discreta de Fourier* de $\mathbf{h}$ y la matriz cuadrada que aparece en (4) se llama la *matriz de Fourier* (de orden n)[1]. A partir de ahora se denotará por $\widehat{\mathbf{h}}$ la transformada discreta de Fourier de $\mathbf{h}$. Si F denota la matriz de Fourier, entonces (3) –o bien (4)– se escribe de forma matricial como

$$\boxed{\mathbf{h} = F\widehat{\mathbf{h}}.} \tag{5}$$

En principio, para que esta definición esté bien hecha, hay que probar que para cada vector $\mathbf{h} \in \mathbb{C}^n$, existe un único vector $\widehat{\mathbf{h}} \in \mathbb{C}^n$ que cumpla (5). Esto se logra si probamos que la matriz F es invertible.

Teorema 2. *Si F es la matriz de Fourier de orden n, entonces $F\overline{F} = nI_n$.*

Demostración: La entrada (r, s) de $F\overline{F}$ es $\sum_{k=1}^{n} F_{rk}\overline{F}_{ks} = \sum_{k=0}^{n-1} \omega^{rk}\overline{\omega}^{ks}$. Como $\omega\overline{\omega} = |\omega|^2 = 1$, entonces $\overline{\omega} = \omega^{-1}$, y por tanto,

$$\sum_{k=0}^{n-1} \omega^{rk}\overline{\omega}^{ks} = \sum_{k=0}^{n-1} \omega^{rk}\omega^{-ks} = \sum_{k=0}^{n-1} \left(\omega^{r-s}\right)^k.$$

Si $r = s$, es evidente que $\omega^{r-s} = 1$ y por tanto, la entrada (r, r) de $F\overline{F}$ es n. Supongamos ahora que $r \neq s$ y denotemos $\xi = \omega^{r-s}$. Como $r - s \neq 0$ y $r, s \in \{1, \cdots, n\}$, entonces $\xi \neq 1$. Pero es más, $\xi^n = 1$ y por tanto,

$$\sum_{k=0}^{n-1} \left(\omega^{r-s}\right)^k = 1 + \xi + \xi^2 + \cdots + \xi^{n-1} = \frac{1 - \xi^n}{1 - \xi} = 0.$$

Por tanto, si $r \neq s$, la entrada (r, s) de $F\overline{F}$ es cero. □

Obviamente, de este teorema, ya que F es una matriz cuadrada, se deduce que $F^{-1} = \frac{1}{n}\overline{F}$, lo que permite hallar la transformada discreta de Fourier de $\mathbf{h} \in \mathbb{C}^n$ sin resolver ningún sistema de ecuaciones, ni invertir ninguna matriz:

$$\boxed{\widehat{\mathbf{h}} = \frac{1}{n}\overline{F}\mathbf{h}.} \tag{6}$$

[1] Hay muchas definiciones distintas de la transformada discreta de Fourier; pero todas ellas se diferencian en pequeños detalles, como la aparición de los factores $1/n$ o $1/\sqrt{n}$ en (5) o bien cambiar ω por $\overline{\omega}$ en la matriz de Fourier.

Obsérvese que aunque sea $\mathbf{h} \in \mathbb{R}^n$, $\widehat{\mathbf{h}}$ puede tener parte imaginaria no nula. De hecho, en muchas aplicaciones prácticas, la señal original es real; por lo que conviene caracterizar esta situación.

Teorema 3. *Sea* $\mathbf{h} \in \mathbb{C}^n$. *Entonces* $\mathbf{h} \in \mathbb{R}^n$ *si y solamente si* $\widehat{\mathbf{h}}[k] = \overline{\widehat{\mathbf{h}}[n-k]}$ *para* $k = 1, \ldots, n-1$ *y* $\widehat{\mathbf{h}}[0] \in \mathbb{R}$.

Demostración: Es bastante sencilla si se usa (5) para una implicación y (6) para la recíproca. □

La siguiente igualdad, donde $\|\cdot\|$ indica la norma euclídea, es importante porque de alguna manera dice que la transformada discreta de Fourier preserva la energía.

$$\|\mathbf{h}\| = \sqrt{n}\|\widehat{\mathbf{h}}\|. \tag{7}$$

Ya que $\|\mathbf{h}\|^2 = \mathbf{h}^*\mathbf{h} = (F\widehat{\mathbf{h}})^* F\widehat{\mathbf{h}} = \widehat{\mathbf{h}}^* F^* F\widehat{\mathbf{h}}$. Como F es simétrica y cumple $\overline{F}F = nI_n$, entonces $F^*F = nI_n$. Por tanto, $\|\mathbf{h}\|^2 = n\|\widehat{\mathbf{h}}\|^2$.

Se ha de decir que para calcular la transformada discreta de Fourier no se usa directamente (6), sino que se usa la llamada *transformada rápida de Fourier*. Esto no es otra transformada, sino que es un algoritmo que permite calcular $\widehat{\mathbf{h}}$ a partir de $\mathbf{h}$ de un modo mucho más eficiente: mientras que usar (6) requiere del orden de n^2 multiplicaciones escalares, la transformada rápida de Fourier solo requiere del orden de $(n/2)\log_2 n$ multiplicaciones escalares. Hay muchos libros en donde se explica la transformada rápida de Fourier, entre los cuales se puede citar [2].

Veamos la interpretación física de la transformada discreta de Fourier. Sea $\mathbf{h} \in \mathbb{R}^n$ una señal discreta y $\widehat{\mathbf{h}} \in \mathbb{C}^n$ su transformada discreta de Fourier. Debido a (3), cada $\widehat{\mathbf{h}}[k]$ muestra cuál es el comportamiento de la oscilación

$$\cos\left(\frac{2\pi k}{n} r\right) + \mathrm{i}\,\mathrm{sen}\left(\frac{2\pi k}{n} r\right), \qquad r = 0, 1, \ldots, n-1. \tag{8}$$

Por eso, se dice que el dominio de $\mathbf{h}$ es el temporal y el de $\widehat{\mathbf{h}}$ es el de las frecuencias. La magnitud $|\widehat{\mathbf{h}}[k]|^2$ es una medida de la energía de la componente (8) de la señal.

3. Filtrado de señales, compresión digital y la transformada discreta de Fourier

Comencemos con un ejemplo (cocinado previamente con ayuda del ordenador).

Ejemplo 1. Consideremos una señal, de la cual se han tomado 51 muestras equiespaciadas como muestra la figura 4.

Esta señal $\mathbf{h}$ es un vector de $\mathbb{R}^{51}$. Como las componentes de su transformada discreta de Fourier son complejas, para representar visualmente $\widehat{\mathbf{h}} = [\widehat{\mathbf{h}}[0], \ldots, \widehat{\mathbf{h}}[50]]^t$, representamos en la figura 5 los valores $|\widehat{\mathbf{h}}[0]|, \ldots, |\widehat{\mathbf{h}}[50]|$.

Vemos que en $\widehat{\mathbf{h}}$ hay dos valores más altos que los demás: el tercero y el penúltimo. Podemos pensar que estos dos valores son los "de verdad", mientras que el resto

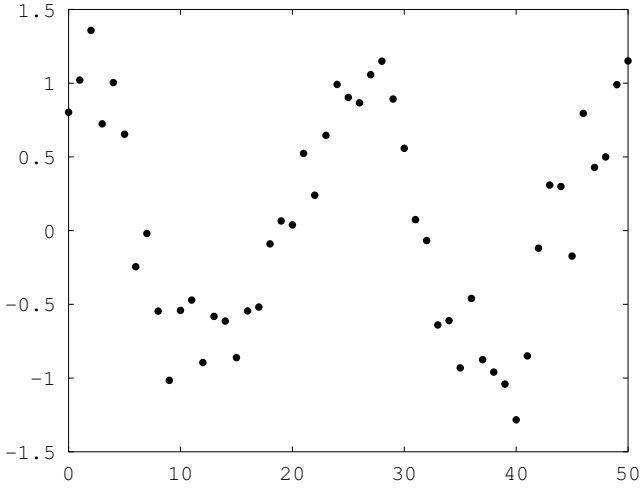

Figura 4: Una señal con ruido.

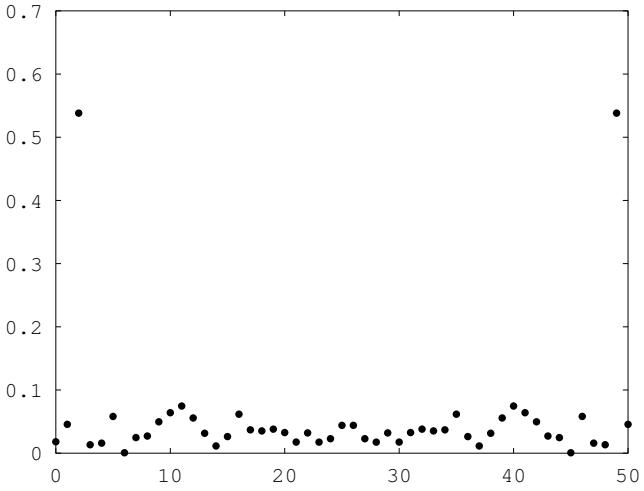

Figura 5: El valor absoluto de la transformada de Fourier discreta de una señal con ruido.

corresponde al ruido de fondo. Vamos a conservar solo estos valores más altos, es decir definimos $\widehat{\mathbf{g}} = [0, 0, \widehat{\mathbf{h}}[2], 0, \ldots, 0, \widehat{\mathbf{h}}[49], 0]^t$.

Por último, volvemos al "dominio temporal". Hacemos $\mathbf{g} = F\widehat{\mathbf{g}}$. Cuidado: $\mathbf{g}$ puede ser un vector complejo (no real); pero por el Teorema 3, resulta que $\mathbf{g}$ es real, estando dibujada esta señal en la figura 6. Vemos en este caso que se trata de una sinusoidal casi perfecta.

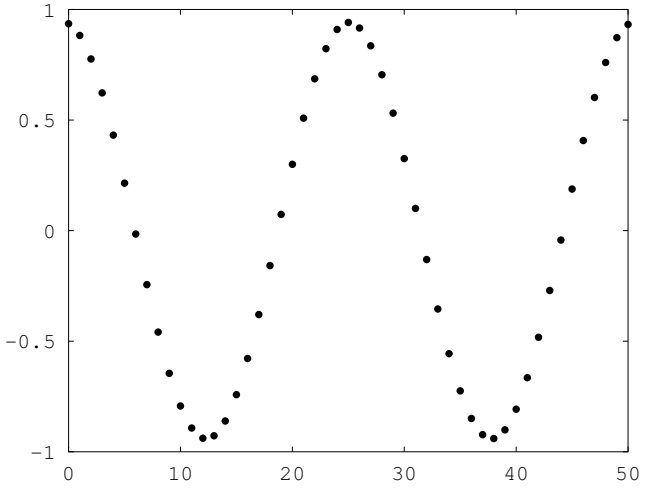

Figura 6: La señal limpiada.

Podemos intentar averiguar la expresión de esta sinusoidal. Para ello, en primer lugar tenemos que saber si es más sencillo manejar $\mathbf{g}$ ó $\widehat{\mathbf{g}}$. Desde luego, es más sencillo $\widehat{\mathbf{g}}$ ya que solo tiene dos componentes no nulas (la tercera y la penúltima). Además podemos ver en la gráfica que $|\widehat{\mathbf{h}}[2]| \simeq |\widehat{\mathbf{h}}[49]|$, como corresponde a un vector $\mathbf{h}$ real. De hecho, con los datos numéricos usados en este ejemplo se tiene que $\widehat{\mathbf{h}}[2] = 0'518 + 0'084\mathrm{i}$ y $\widehat{\mathbf{h}}[49] = 0'518 - 0'084\mathrm{i}$. Llamemos ahora $a = 0'518$ y $b = 0'084$. Si usamos (3) tenemos, ya que $n = 51$,

$$\begin{aligned}\mathbf{g}[k] &= \sum_{r=0}^{50} \widehat{\mathbf{g}}[r]\mathrm{e}^{2\pi r\mathrm{i}k/51} = \widehat{\mathbf{h}}[2]\mathrm{e}^{2\pi 2\mathrm{i}k/51} + \widehat{\mathbf{h}}[49]\mathrm{e}^{2\pi 49\mathrm{i}k/51} \\ &= (a+b\mathrm{i})\mathrm{e}^{4\pi\mathrm{i}k/51} + (a-b\mathrm{i})\mathrm{e}^{98\pi\mathrm{i}k/51}.\end{aligned}$$

Ya que $\mathrm{e}^{98\pi\mathrm{i}k/51} = \mathrm{e}^{-4\pi\mathrm{i}k/51}$, entonces

$$\begin{aligned}\mathbf{g}[k] &= (a+b\mathrm{i})\mathrm{e}^{4\pi\mathrm{i}k/51} + (a-b\mathrm{i})\mathrm{e}^{-4\pi\mathrm{i}k/51} \\ &= a(\mathrm{e}^{4\pi\mathrm{i}k/51} + \mathrm{e}^{-4\pi\mathrm{i}k/51}) + b\mathrm{i}(\mathrm{e}^{4\pi\mathrm{i}k/51} - \mathrm{e}^{-4\pi\mathrm{i}k/51}) \\ &= 2a\cos(4\pi k/51) - 2b\,\mathrm{sen}(4\pi k/51) \\ &= 1'036\cos(4\pi k/51) - 0'168\,\mathrm{sen}(4\pi k/51).\end{aligned}$$

Pensemos en este ejemplo: Se parte de una señal discreta $\mathbf{h}$. Se calcula su transformada discreta de Fourier obteniéndose $\widehat{\mathbf{h}}$. Se modifica esta transformada, logrando

$\widehat{\mathbf{g}}$. Y por último se antitransforma $\widehat{\mathbf{g}}$ consiguiendo $\mathbf{g}$.

$$\begin{array}{ccc} \mathbf{h} & \xrightarrow{\text{Transformada de Fourier}} & \widehat{\mathbf{h}} \\ & & \Big\downarrow \text{Filtrado} \\ \mathbf{g} & \xleftarrow{\text{Antitransformada de Fourier}} & \widehat{\mathbf{g}} \end{array}$$

La pregunta es: ¿se parecen $\mathbf{h}$ y $\mathbf{g}$? O dicho de otro modo: si el cambio que hago en $\widehat{\mathbf{h}}$ para conseguir $\widehat{\mathbf{g}}$ es pequeño, ¿se parecerán $\mathbf{h}$ y $\mathbf{g}$?

Resulta que (7) da una respuesta bastante precisa sobre esta pregunta, ya que aplicando (7) para $\mathbf{g}-\mathbf{h}$ (en vez de $\mathbf{h}$) se tiene

$$\|\mathbf{g}-\mathbf{h}\| = \sqrt{n}\|\widehat{\mathbf{g}-\mathbf{h}}\| = \sqrt{n}\|\widehat{\mathbf{g}}-\widehat{\mathbf{h}}\|. \tag{9}$$

Esto quiere decir, entre otras cosas, que si $\|\widehat{\mathbf{g}}-\widehat{\mathbf{h}}\|$ es "pequeño", entonces $\|\mathbf{g}-\mathbf{h}\|$ es también "pequeño". Aparentemente, el factor $\sqrt{n}$ es desagradable, ya que si n es muy grande, entonces $\|\mathbf{g}-\mathbf{h}\|$ puede llegar a ser muy grande. Pero en realidad, esto no es así, ya que todo depende de cómo se define la transformada discreta de Fourier.

Ejemplo 2. Veamos un ejemplo de la utilidad de (9). Consideremos la señal discreta $\mathbf{g} = [1, 2, 3, 3'5, 3, 2, 1]^t \in \mathbb{R}^7$. Tras aplicar la transformada de Fourier discreta a $\mathbf{g}$ obtenemos

$$\begin{aligned}\widehat{\mathbf{g}} = [2'214, \quad & -0'585 - 0'282\mathrm{i}, \quad -0'017 - 0'021\mathrm{i}, \quad -0'005 - 0'02\mathrm{i}, \\ & -0'005 + 0'02\mathrm{i}, \quad -0'017 + 0'021\mathrm{i}, \quad -0'585 + 0'282\mathrm{i}]^t \in \mathbb{C}^7.\end{aligned}$$

La situación se observa mejor si formamos el vector de $\mathbb{R}^7$ cuya k-ésima componente es $|\widehat{\mathbf{g}}[k]|$:

$$[2'21, \quad 0'650, \quad 0'027, \quad 0'020, \quad 0'020, \quad 0'027, \quad 0'650].$$

Vemos que la tercera, cuarta, quinta y sexta componentes de $\widehat{\mathbf{g}}$ son pequeñas en comparación con el resto. Vamos a anular estas componentes formando

$$\widehat{\mathbf{h}} = [2'214, \quad -0'585 - 0'282\mathrm{i}, \quad 0, \quad 0, \quad 0, \quad 0, \quad -0'585 + 0'282\mathrm{i}]^t \in \mathbb{C}^7.$$

Y ahora volvemos para atrás, recuperando $\mathbf{h}$:

$$\mathbf{h} = [1'043, \quad 1'924, \quad 3'024, \quad 3'514, \quad 3'025, \quad 1'925, \quad 1'043]^t.$$

Podemos observar que $\mathbf{g}$ y $\mathbf{h}$ son realmente parecidos.

Y... ¿qué utilidad tiene esto? Podemos ver que mientras $\widehat{\mathbf{g}}$ tiene 7 entradas no nulas, el vector $\widehat{\mathbf{h}}$ tiene solo 3 entradas no nulas, pero además la segunda y última componentes de $\widehat{\mathbf{h}}$ son conjugadas una de la otra; es decir, hemos comprimido en más de un 50 % la señal original $\mathbf{g}$ (perdiendo una mínima información). Además podemos saber de manera precisa la distorsión $\|\mathbf{g}-\mathbf{h}\|$ usando (9). Observemos que

para que este proceso de compresión sea efectivo en el sentido de que $\mathbf{h} \simeq \mathbf{g}$, hemos de asegurarnos de que $\widehat{\mathbf{h}} \simeq \widehat{\mathbf{g}}$; es decir, las componentes de $\widehat{\mathbf{h}}$ que desechemos tienen que ser "pequeñas".

Ejemplo 3. Consideremos la señal $\mathbf{g} = [1, 2, 3, 4, 5, 6, 7]^t \in \mathbb{R}^7$. ¡Más regular no puede ser! Su transformada de Fourier discreta es

$$\begin{aligned}\widehat{\mathbf{g}} = [4,\ &-0'5 + 1'038\mathrm{i},\ -0'5 + 0{,}399\mathrm{i},\\ &-0'5 + 0'114\mathrm{i},\ -0'5 - 0'114\mathrm{i},\ -0{,}5 - 0{,}399\mathrm{i}, -0'5 - 1'038\mathrm{i}]^t,\end{aligned}$$

vemos ninguna componente de $\widehat{\mathbf{g}}$ es pequeña. Este hecho se ve mejor si calculamos el vector cuyas componentes es el valor absoluto de las componentes de $\widehat{\mathbf{g}}$:

$$\left[\ 4,\quad 1'152,\quad 0'64\quad 0'513,\quad 0'513,\quad 0'639,\quad 1'152\ \right]^t.$$

Podríamos pensar en anular la cuarta y quinta componente de $\widehat{\mathbf{g}}$. Sea

$$\widehat{\mathbf{h}} = \left[\ 4,\quad -0'5 + 1'038\mathrm{i},\quad -0'5 + 0'399\mathrm{i},\quad 0,\quad 0,\quad -0'5 - 0'399\mathrm{i},\quad -0'5 - 1'038\mathrm{i}\ \right]^t.$$

Si calculamos la antitransformada de $\widehat{\mathbf{h}}$ logramos

$$\mathbf{h} = \left[\ 2,\quad 1'198,\quad 3'445,\quad 4,\quad 4'555,\quad 6'802,\quad 6\ \right]^t.$$

Vemos ahora que $\mathbf{g}$ y $\mathbf{h}$ no se parecen mucho. De hecho, esto ya lo podríamos haber establecido usando (9). Como en este ejemplo, $\widehat{\mathbf{g}}$ y $\widehat{\mathbf{h}}$ no son parecidos; de hecho se tiene $\|\widehat{\mathbf{g}} - \widehat{\mathbf{h}}\| = 0'725$, entonces sin haber calculado $\mathbf{h}$, se tiene por (9) que $\|\mathbf{g} - \mathbf{h}\| = 1'919$; un valor que no es pequeño.

¿Por qué estos dos ejemplos tienen un comportamiento tan distinto? La situación se aclara si pensamos que las señales son *periódicas*. La señal del ejemplo 3 debe ser

$$\left[\ \cdots\quad 5,\quad 6,\quad 7,\quad 1,\quad 2,\quad 3,\quad 4,\quad 5,\quad 6,\quad 7,\quad 1,\quad 2,\quad \cdots\ \right]$$

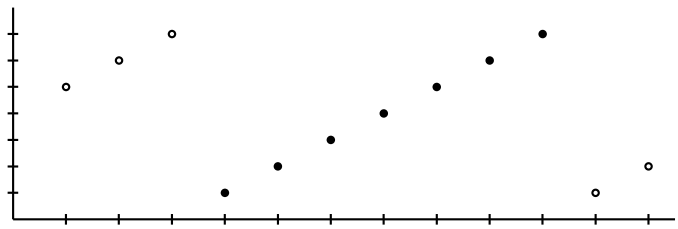

Figura 7: Los puntos en negro son la señal finita; pero en realidad la señal es infinita. Obsérvese que hay "saltos".

Esta es la razón de que la señal del ejemplo 3 se "comporte mal" cuando hallamos su transformada discreta de Fourier. En realidad, las palabras "mal comportamiento" son erróneas: de hecho se comporta como debería comportarse: debido a que la señal tiene saltos considerables, las altas frecuencias son significativas.

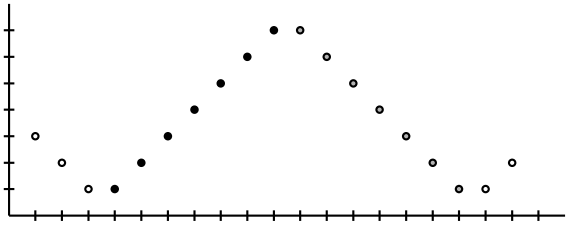

Figura 8: Extensión simétrica de la señal finita.

Pero si solo estamos interesados en el tramo finito (en la figura 7, los puntos marcados en negro), hay una manera de evitar estos saltos: simetrizar la señal. La mejor manera es verlo en la gráfica 8.

Ejemplo 4. Sea ahora la señal

$$\mathbf{g} = \left[\begin{array}{cccccccccccccc} 1, & 2, & 3, & 4, & 5, & 6, & 7, & 7, & 6, & 5, & 4, & 3, & 2, & 1 \end{array}\right]^t \in \mathbb{R}^{14}.$$

Escribimos ahora $\widehat{\mathbf{g}}$ (con una precisión de 10^{-3}):

$$\begin{array}{l} \widehat{\mathbf{g}} = \left[\begin{array}{cccccccc} 4 & -1'371 - 0'313\mathrm{i} & 0 & -0'112 - 0'09\mathrm{i} & 0 & -0'017 - 0'034\mathrm{i} & 0 \\ 0, & 0, & -0'017 + 0'034\mathrm{i}, & 0, & -0'112 + 0'09\mathrm{i}, & 0, & -1'371 + 0'313\mathrm{i} \end{array}\right]. \end{array} \tag{10}$$

Hacemos nulas dos componentes más de $\widehat{\mathbf{g}}$:

$$\begin{array}{rl} \widehat{\mathbf{h}} = \left[\right. & 4, \quad -1'371 - 0'313\mathrm{i}, \quad 0, \quad -0'112 - 0'089\mathrm{i}, \quad 0, \quad 0, \quad 0, \\ = & 0, \quad 0, \quad 0, \quad 0, \quad -0'112 + 0'089\mathrm{i}, \quad 0, \quad -1'371 + 0'313\mathrm{i} \left.\right]. \end{array}$$

De esta manera conseguimos el vector $\widehat{\mathbf{h}}$ (muy parecido a $\widehat{\mathbf{g}}$). Y por último, antitransformamos $\widehat{\mathbf{h}}$ logrando

$$\begin{array}{l} \mathbf{h} = \left[\begin{array}{ccccccccc} 1'03, & 1'93, & 3'06, & 4, & 4'94, & 6'07, & 6'97, \\ 6'97, & 6'07, & 4{,}94, & 4, & 3'06, & 1'93, & 1'03 \end{array}\right]. \end{array} \tag{11}$$

Ahora sí hay un parecido entre $\mathbf{g}$ y $\mathbf{h}$.

Vamos a contar el "ahorro de almacenaje". La señal original (sin simetrizar) tiene 7 entradas reales. La extensión simétrica de la señal tiene 14 entradas. Anulamos 9 entradas de $\widehat{\mathbf{g}}$ en (10), quedándonos con 5 entradas no nulas; pero complejas. Pero, observe que en (11) hay componentes que son conjugadas. Por tanto, para almacenar $\widehat{\mathbf{h}}$ hacen falta 5 números reales.

4. La transformada discreta coseno

Sea $\mathbf{h} = [\mathbf{h}[0], \mathbf{h}[1], \cdots, \mathbf{h}[n-1]]^t \in \mathbb{R}^n$. En primer lugar definimos la extensión simétrica[2] de $\mathbf{h}$ por medio de

$$\mathbf{y} = \left[\begin{array}{ccccccccc} \mathbf{h}[0], & \mathbf{h}[1], & \cdots, & \mathbf{h}[n-1], & \mathbf{h}[n-1], & \cdots, & \mathbf{h}[1], & \mathbf{h}[0] \end{array}\right]^t \in \mathbb{R}^{2n}. \tag{12}$$

[2]Dependiendo de cómo hagamos la extensión, obtendremos distintas transformadas discretas de coseno. De hecho, según la Wikipedia hay cuatro transformadas discretas de coseno distintas.

Ahora aplicamos la transformada de Fourier discreta a $\mathbf{y}$. Sea

$$\widehat{\mathbf{y}} = \left[\ \widehat{\mathbf{y}}[0],\quad \widehat{\mathbf{y}}[1],\quad \cdots,\quad \widehat{\mathbf{y}}[2n-1]\ \right]^t \in \mathbb{C}^{2n}$$

la transformada discreta de Fourier de $\mathbf{y}$; es decir

$$\widehat{\mathbf{y}} = \frac{1}{2n}\overline{F}\mathbf{y},$$

donde la matriz F está definida ahora (recuérdese que estamos calculando la transformada de Fourier para vectores de $2n$ componentes y no de n) como

$$F_{rs} = \omega^{rs}, \quad r,s = 0,\ldots,2n-1, \qquad \omega = \exp\left(\frac{2\pi\mathrm{i}}{2n}\right) = \exp\left(\frac{\pi\mathrm{i}}{n}\right).$$

Puesto que aparece $\overline{F}$, denotaremos $\xi = \overline{\omega} = \exp(-\pi j/n)$. Sea $k = 0,1,\ldots,2n-1$.

$$\widehat{\mathbf{y}}[k] = \frac{1}{2n}(\text{fila } k \text{ de } \overline{F})\ \mathbf{y}$$

$$= \frac{1}{2n}\left[\ \xi^0 \quad \xi^k \quad \cdots \quad \xi^{k(n-1)} \quad \xi^{kn} \quad \cdots \quad \xi^{k(2n-2)} \quad \xi^{k(2n-1)}\ \right] \begin{bmatrix} \mathbf{h}[0] \\ \mathbf{h}[1] \\ \vdots \\ \mathbf{h}[n-1] \\ \mathbf{h}[n-1] \\ \vdots \\ \mathbf{h}[1] \\ \mathbf{h}[0] \end{bmatrix}$$

$$= \frac{1}{2n}\left[(\xi^0 + \xi^{k(2n-1)})\mathbf{h}[0] + (\xi^k + \xi^{k(2n-2)})\mathbf{h}[1] + \cdots + (\xi^{k(n-1)} + \xi^{kn})\mathbf{h}[n-1]\right].$$

El coeficiente de $\mathbf{h}[r]$ es $\xi^{kr} + \xi^{k(2n-1-r)}$. Vamos a ver si simplificamos este coeficiente. Recordemos que $\xi = \exp(-\pi\mathrm{i}/n)$ y por tanto $\xi^n = -1$ y $\xi^{2n} = 1$.

$$\begin{aligned}
\xi^{kr} + \xi^{k(2n-1-r)} &= \xi^{kr} + \xi^{-k(1+r)} \\
&= \xi^{-k/2}\left(\xi^{kr+k/2} + \xi^{-kr-k/2}\right) \\
&= \xi^{-k/2}\left(\exp\left(\frac{-\pi\mathrm{i}(kr+k/2)}{n}\right) + \exp\left(\frac{\pi\mathrm{i}(kr+k/2)}{n}\right)\right) \\
&= 2\xi^{-k/2}\cos\left(\frac{\pi(kr+k/2)}{n}\right) = 2\omega^{k/2}\cos\left(\frac{\pi k(2r+1)}{2n}\right).
\end{aligned}$$

Por tanto,

$$\widehat{\mathbf{y}}[k] = \frac{\omega^{k/2}}{n}\ \sum_{r=0}^{n-1}\mathbf{h}[r]\cos\left(\frac{\pi k(2r+1)}{2n}\right). \tag{13}$$

Esta expresión permite probar que $\widehat{\mathbf{y}}[n] = 0$ ya que $\cos(m\pi/2) = 0$ para $m \in \mathbb{N}$ impar. Además motiva la definición de la transformada discreta coseno (la aparición de $\sqrt{n}$ y $\sqrt{2}$ son simplemente debidos a la costumbre).

Definición. La *transformada discreta coseno* de un vector $\mathbf{h} \in \mathbb{R}^n$ es $C\mathbf{h}$, donde la matriz C está definida por medio de

$$C = \frac{1}{\sqrt{n}} \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \sqrt{2}\cos\theta & \sqrt{2}\cos(3\theta) & \cdots & \sqrt{2}\cos((2n-1)\theta) \\ \vdots & \vdots & \ddots & \vdots \\ \sqrt{2}\cos((n-1)\theta) & \sqrt{2}\cos(3(n-1)\theta) & \cdots & \sqrt{2}\cos((2n-1)(n-1)\theta) \end{bmatrix}, \tag{14}$$

y $\theta = \pi/(2n)$.

Recordemos que la matriz F de la transformada discreta de Fourier cumple $F\overline{F} = nI_n$. ¿Cumplirá algo similar la matriz C definida en (14)? La relación $F\overline{F} = nI_n$ es muy importante, ya que permite calcular la antitransformada discreta de Fourier *sin tener que calcular la inversa de ninguna matriz.*

Sea $\mathbf{c} \in \mathbb{R}^n$ y queremos saber si existe y cómo se puede calcular $\mathbf{h} \in \mathbb{R}^n$ de modo que $C\mathbf{h} = \mathbf{c}$. Suponiendo que exista tal $\mathbf{h}$, definimos $\mathbf{y} \in \mathbb{R}^{2n}$ por medio de (12). Sea $\widehat{\mathbf{y}} \in \mathbb{R}^{2n}$ la transformada discreta de Fourier de $\mathbf{y}$. Como $\mathbf{y} = F\widehat{\mathbf{y}}$, entonces para $k = 0, \ldots, n-1$

$$\begin{aligned}\mathbf{h}[k] = {} & \omega^0\ \widehat{\mathbf{y}}[0] + \omega^k\ \widehat{\mathbf{y}}[1] + \cdots + \omega^{k(n-1)}\ \widehat{\mathbf{y}}[n-1] + \\ & + \omega^{kn}\ \widehat{\mathbf{y}}[n] + \omega^{k(n+1)}\ \widehat{\mathbf{y}}[n+1] + \omega^{k(n+2)}\ \widehat{\mathbf{y}}[n+2] + \cdots + \omega^{k(2n-1)}\ \widehat{\mathbf{y}}[2n-1].\end{aligned}$$

Ahora es cuando podemos aprovechar el hecho de que la transformada de Fourier discreta de señales reales presenta la "simetría" señalada en el Teorema 3. Como $\widehat{\mathbf{y}}[n] = 0$ e $\overline{\widehat{\mathbf{y}}[s]} = \widehat{\mathbf{y}}[2n-s]$ para $s = 1, \ldots, n-1$, agruparemos $\widehat{\mathbf{y}}[s]$ con $\widehat{\mathbf{y}}[2n-s]$ de la forma siguiente:

$$\begin{aligned}\mathbf{h}[k] = {} & \widehat{\mathbf{y}}[0] \\ & + \left[\omega^k\ \widehat{\mathbf{y}}[1] + \omega^{k(2n-1)}\ \widehat{\mathbf{y}}[2n-1]\right] + \cdots + \left[\omega^{k(n-1)}\ \widehat{\mathbf{y}}[n-1] + \omega^{k(n+1)}\ \widehat{\mathbf{y}}[n+1]\right].\end{aligned}$$

Debido a que $\omega = \exp(\pi\mathrm{i}/n)$ se tiene

$$\begin{aligned}\omega^{kr}\ \widehat{\mathbf{y}}[r] + \omega^{k(2n-r)}\ \widehat{\mathbf{y}}[2n-r] &= \omega^{kr}\ \widehat{\mathbf{y}}[r] + \omega^{-kr}\ \overline{\widehat{\mathbf{y}}[r]} \\ &= \omega^{kr}\ \widehat{\mathbf{y}}[r] + \overline{\omega^{kr}\ \widehat{\mathbf{y}}[r]} = 2\mathrm{Re}\left(\omega^{kr}\ \widehat{\mathbf{y}}[r]\right),\end{aligned}$$

como $\widehat{\mathbf{y}}[0]$ es real (esto se deduce de (13)), se tiene

$$\mathbf{h}[k] = \mathrm{Re}\left(\widehat{\mathbf{y}}[0] + 2\omega^k\ \widehat{\mathbf{y}}[1] + \cdots + 2\omega^{(n-1)k}\ \widehat{\mathbf{y}}[n-1]\right). \tag{15}$$

Ahora pondremos cada $\widehat{\mathbf{y}}[r]$ en función de $\mathbf{c}$ para poder expresar (usando (15)) cada $\mathbf{h}[k]$ en términos de $\mathbf{c}$. Definimos $\mathbf{b} = [\widehat{\mathbf{y}}[0], \widehat{\mathbf{y}}[1], \cdots, \widehat{\mathbf{y}}[n-1]]^t \in \mathbb{C}^n$ (obsérvese que

b es la "mitad" de $\widehat{\mathbf{y}}$). Por (13) se tiene que si definimos

$$W = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & \omega^{1/2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \omega^{(n-1)/2} \end{bmatrix}$$

y

$$C_1 = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \cos\theta & \cos(3\theta) & \cdots & \cos((2n-1)\theta) \\ \vdots & \vdots & \ddots & \vdots \\ \cos((n-1)\theta) & \cos(3(n-1)\theta) & \cdots & \cos((2n-1)(n-1)\theta) \end{bmatrix},$$

entonces

$$\mathbf{b} = \frac{1}{n} W C_1 \mathbf{h}.$$

Pero, como es fácil comprobar, debido a la definición de la matriz C hecha en (14),

$$C = \frac{1}{\sqrt{n}} \underbrace{\begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & \sqrt{2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sqrt{2} \end{bmatrix}}_{:=D} C_1.$$

Por tanto,

$$\mathbf{b} = \frac{1}{n} W C_1 \mathbf{h} = \frac{1}{n} W \left(\sqrt{n} D^{-1} C\right) \mathbf{h} = \frac{1}{\sqrt{n}} W D^{-1} C \mathbf{h} = \frac{1}{\sqrt{n}} W D^{-1} \mathbf{c}.$$

Ya hemos puesto **b** en función de **c**. Retomamos los cálculos hechos en (15).

$$\begin{aligned} \widehat{\mathbf{y}}[0] + 2\omega^k\, \widehat{\mathbf{y}}[1] + \cdots + 2\omega^{(n-1)k}\, \widehat{\mathbf{y}}[n-1] = \\ &= \begin{bmatrix} 1 & 2\omega^k & \cdots & 2\omega^{k(n-1)} \end{bmatrix} \begin{bmatrix} \widehat{\mathbf{y}}[0] \\ \widehat{\mathbf{y}}[1] \\ \vdots \\ \widehat{\mathbf{y}}[n-1] \end{bmatrix} \\ &= \begin{bmatrix} 1 & 2\omega^k & \cdots & 2\omega^{k(n-1)} \end{bmatrix} \mathbf{b} \\ &= \begin{bmatrix} 1 & 2\omega^k & \cdots & 2\omega^{k(n-1)} \end{bmatrix} \frac{1}{\sqrt{n}} W D^{-1} \mathbf{c} \\ &= \frac{1}{\sqrt{n}} \begin{bmatrix} 1 & \sqrt{2}\omega^{k+1/2} & \cdots & \sqrt{2}\omega^{k(n-1)+(n-1)/2} \end{bmatrix} \mathbf{c}. \end{aligned}$$

No perdamos de vista que en (15) aparece la parte real de

$$\widehat{\mathbf{y}}[0] + 2\omega^k\, \widehat{\mathbf{y}}[1] + \cdots + 2\omega^{(n-1)k}\, \widehat{\mathbf{y}}[n-1]$$

y que $\mathbf{c}$ es una señal real. Por tanto, vamos a tomar la parte real de $\omega^{kr+r/2}$, para $r = 1, \ldots, n-1$. Asimismo recuerde que $\omega = \exp(\pi \mathrm{i}/n)$ y $\theta = \pi/(2n)$.

$$\begin{aligned}\omega^{kr+r/2} &= \exp\left(\left(kr + \frac{r}{2}\right)\frac{\pi \mathrm{i}}{n}\right) \\ &= \exp\left(\pi \mathrm{i}\frac{2kr+r}{2n}\right) = \cos\left((2k+1)r\theta\right) + \mathrm{i}\,\mathrm{sen}\left((2k+1)r\theta\right).\end{aligned}$$

Por tanto,

$$\begin{aligned}\mathbf{h}[k] &= \mathrm{Re}\left(\widehat{\mathbf{y}}[0] + 2\omega^{k}\,\widehat{\mathbf{y}}[1] + \cdots + 2\omega^{(n-1)k}\,\widehat{\mathbf{y}}[n-1]\right) \\ &= \frac{1}{\sqrt{n}}\left[\begin{array}{cccc} 1 & \sqrt{2}\cos((2k+1)\theta) & \cdots & \sqrt{2}\cos((2k+1)(n-1)\theta) \end{array}\right]\mathbf{c}.\end{aligned}$$

De forma matricial

$$\mathbf{h} = \frac{1}{\sqrt{n}}\begin{bmatrix} 1 & \sqrt{2}\cos\theta & \cdots & \sqrt{2}\cos(\theta(n-1)) \\ 1 & \sqrt{2}\cos(3\theta) & \cdots & \sqrt{2}\cos(3(n-1)\theta) \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \sqrt{2}\cos((2n-1)\theta) & \cdots & \sqrt{2}\cos((2n-1)(n-1)\theta) \end{bmatrix}\mathbf{c}. \qquad (16)$$

Resumamos lo que hemos hecho hasta ahora: dada una señal $\mathbf{c} \in \mathbb{R}^n$, hemos demostrado que si existe $\mathbf{h} \in \mathbb{R}^n$ tal que $C\mathbf{h} = \mathbf{c}$, entonces tal $\mathbf{h}$ es única y viene forzosamente dada por (16). Es decir, hemos encontrado la expresión para la transformada coseno discreta inversa[3].

Resulta que la matriz cuadrada que aparece en (16) es ¡la traspuesta de la matriz C que aparece en (14)! Esto permite calcular la inversa de la transformada coseno discreta sin invertir ninguna matriz. Por lo que si $\mathbf{h} \in \mathbb{R}^n$ es una señal y $\mathbf{c} \in \mathbb{R}^n$ es la trasformada coseno discreta de $\mathbf{h}$, entonces

$$\boxed{C\mathbf{h} = \mathbf{c}, \qquad C^t\mathbf{c} = \mathbf{h},}$$

donde la matriz C está definida en (14). De estas dos últimas igualdades se obtiene $CC^t = C^tC = I_n$, lo que también permite demostrar de forma evidente que $\|\mathbf{c}\| = \|\mathbf{h}\|$.

5. La transformada discreta coseno bidimensional

La idea fundamental del sistema de compresión JPEG es aplicar la transformada discreta coseno a una matriz (y no a un vector) ya que una imagen se puede modelar como una matriz. Por tanto, en esta sección vamos a ver cómo se puede definir de forma natural la transformada de una matriz.

[3]Hemos demostrado la unicidad de la solución de $C\mathbf{h} = \mathbf{c}$, pero no la existencia. Pero un resultado de álgebra matricial establece que si una matriz A es cuadrada, entonces el sistema $A\mathbf{x} = \mathbf{0}$ tiene solución única si y solamente A es invertible. Por tanto, la matriz C es invertible y $\mathbf{h} = C^{-1}\mathbf{c}$.

Por simplificar, vamos a suponer que la matriz que queremos transformar es una matriz cuadrada de tamaño n. De hecho, el primer paso del sistema JPEG es dividir la fotografía en bloques de 8×8 píxeles y aplicar la transformada discreta coseno a cada bloque 8×8. Sea M una matriz $n \times n$. Esta matriz la podemos pensar como un operador de $\mathbb{R}^n$ a $\mathbb{R}^n$ que actúa $\mathbf{x} \mapsto M\mathbf{x}$. Entonces podemos pensar en el diagrama siguiente.

$$\begin{array}{ccc} \mathbb{R}^n & \xrightarrow{C} & \mathbb{R}^n \\ {\scriptstyle M}\downarrow & & \downarrow \text{Transformada de } M \text{ por medio de } C \\ \mathbb{R}^n & \xrightarrow{C} & \mathbb{R}^n \end{array}$$

A la izquierda tenemos los objetos que van a ser transformados por C y a la derecha los objetos que ya han sido transformados por C. Si queremos definir la "transformada de M por medio de C" podemos pensar en el siguiente diagrama y exigir que dé igual ir desde la esquina superior izquierda a la esquina inferior derecha por los dos caminos distintos. Denotaremos a partir de ahora $\mathcal{C}M$ la transformada de la matriz M por medio de C.

$$\begin{array}{ccc} \mathbb{R}^n & \xrightarrow{C} & \mathbb{R}^n \\ & & \downarrow{\scriptstyle \mathcal{C}M} \\ \mathbb{R}^n & & \mathbb{R}^n \end{array} \quad = \quad \begin{array}{ccc} \mathbb{R}^n & & \mathbb{R}^n \\ {\scriptstyle M}\downarrow & & \\ \mathbb{R}^n & \xrightarrow{C} & \mathbb{R}^n \end{array}$$

O con fórmulas,

$$CM = (\mathcal{C}M)\, C.$$

Como C es invertible, podemos despejar $\mathcal{C}M$. Esto motiva la siguiente definición:
Definición.[4] Sea C una matriz invertible de orden n y M una matriz de orden n. Entonces definimos la transformada de M por medio de C como $\mathcal{C}M = CMC^{-1}$.

Volviendo al caso que nos interesa: ¿cómo calcular la transformada coseno discreta de una matriz? Sea M una matriz cuadrada de orden n (que va a ser un bloque cuadrado de la imagen que nos va a interesar manipular). Entonces,

$$\text{La transformada coseno discreta de } M = CMC^{-1} = CMC^t. \tag{17}$$

Obsérvese que $CMC^t = C(CM^t)^t = [C(CM)^t]^t$, lo que significa que la transformada coseno discreta de una matriz M equivale a hacer la transformada coseno discreta de cada fila de M y a continuación, en la matriz resultante, la transformada coseno discreta de cada columna (o en el otro orden). Esto puede ayudar al lector a programar la transformada bidimensional en, por ejemplo, Octave.

Por supuesto, también tenemos que saber dar el paso opuesto: es decir, dada una matriz ya transformada N, ¿cómo hallar la matriz M de la cual proviene? Esto es fácil si de $CM = NC$ despejamos M.

$$\text{La transformada coseno discreta inversa de } N = C^{-1}NC = C^tNC. \tag{18}$$

[4] El lector problamente recuerde cómo son las matrices de una aplicación lineal en bases distintas.

Ejemplo 5. En el siguiente ejemplo veremos un poco cómo se puede manejar Octave (unas modificaciones mínimas sirven para el programa MatLab) para empezar a entender el sistema de compresión JPEG. La siguiente figura (de dominio público) puede descagarse en `http://www.wpclipart.com/recreation/games/chess/`.

Si hemos cargado esta figura con el nombre `rey.jpg`, entonces la orden

```
A = imread("rey.jpg");
```

crea la matriz A en donde se almacena el fichero gráfico. No olvide el punto y coma para que no muestre el resultado (en este ejemplo A es de tamaño 500×500). Como solo hay píxeles blancos y negros, las entradas de A toman el valor 0 (negro) o bien 255 (blanco).

La función `imread` crea una matriz de enteros (que tienen una aritmética más restringida que la de los complejos). Para poder manipular la matriz A, "convertimos las entradas de A a números complejos" (con el comando `double`). Calculamos $\mathcal{C}A$ mediante (17) y almacenamos esta trasformada en la variable `AC`. Si ejecutamos `mean(mean(abs(AC)))` obtenemos $15'187$, lo que nos dice la media del valor absoluto de todas las entradas de `AC`; pero si ejecutamos `max(max(abs(AC)))` obtenemos 103960, lo que nos puede indicar que hay muchas entradas de `AC` muy pequeñas. Con

```
indices = find(abs(AC)<1000);
```

buscamos las entradas de `AC` que son menores en valor absoluto que 1000. Observemos que hay 249708 entradas de este tipo (de un total de $500^2 = 250000$), Anulamos estas entradas (anulamos el 99'883 % de las entradas) y antitransformamos, usando (18), obteniendo una nueva matriz almacenada en la matriz `B`. Por último, "convertimos" `B` a una matriz de enteros con `B=uint8(B)` y creamos el fichero JPEG correspondiente.

```
imwrite(B,"reybis.jpg")
```

En la figura siguiente se han puesto tres ficheros gráficos. El primero es el descrito, en el segundo se anulan las componentes de `AC` menores en valor aboluto que 500 y en el tercero, se anulan las componentes de `AC` menores en valor aboluto que 100. Incluso, si descartamos las entradas de `AC` menores en valor absoluto que 20 logramos

Figura 9: En el primer gráfico se ha logrado una compresión del 99'83 %. En el segundo del 99'76 %. En el tercero del 98'17 %.

una compresión del 90'8 % produciendo una imagen prácticamente indistinguible de la original.

Si ha ejecutado este último ejemplo, podrá observar que la parte de transformar y antitransformar es lenta (y eso que lo hemos hecho para una imagen muy sencilla: solo blancos y negros, sin grises, sin colores y relativamente pequeña). Una idea básica de la compresión JPEG evita la manipulación de matrices grandes. Por fin, describiremos el sistema JPEG en la sección siguiente.

6. El sistema de compresión JPEG

Ya que uno de los pasos básicos del proceso JPEG es anular algunas entradas de la matriz transformada de la imagen, vamos a analizar tales cambios. Y para esto usaremos matrices 4×4.

Denotemos por E_{rs} la matriz de orden 4 cuya entrada (r, s) es 600 y el resto de sus entradas son ceros. Sea G_{rs} la antitransformada coseno discreta de E_{rs}. Un cambio en la entrada (r, s) de la transformada de una imagen se ve reflejada si observamos la imagen asociada a G_{rs} (como los números son pequeños, en la definición de E_{rs} se han elegido 600 en lugar de lo más natural que hubiera sido 1. Este valor de 600 se ha puesto para que las gráficas sean claras). Para generar la primera imagen de la figura 10 se guarda G_{00} en la matriz `a` y se ejecuta

```
imwrite(a,"c00.jpg")
```

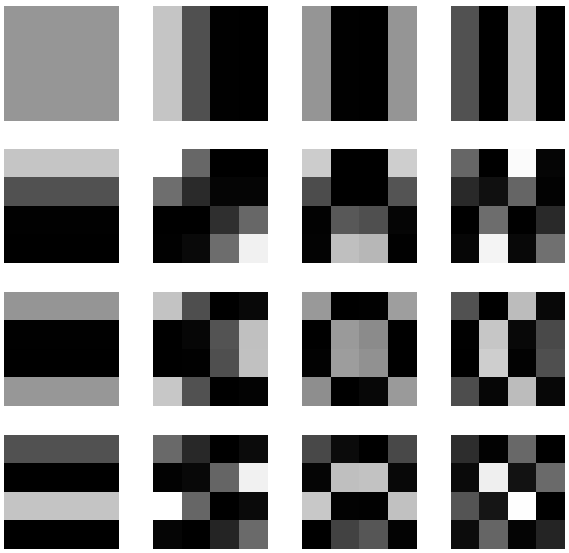

Figura 10: Cualquier imagen 4×4 se puede descomponer como una combinación de estas 16 imágenes.

Quizá algunas fórmulas ayuden a entender tanto la figura 10 como la idea básica de la compresión JPEG. Sea A una matriz cuadrada de tamaño $n \times n$ y $\mathcal{C}A$ su transformada coseno discreta bidimensional. Esta transformada se puede expresar como

$$\mathcal{C}A = \sum_{r,s} \lambda_{rs} E_{rs},$$

siendo E_{rs} ahora la matriz cuadrada con un sólo uno en la posición (r, s) y el resto de sus entradas nulas (salvo un múltiplo, son las E_{rs} definidas anteriormente). Observe que λ_{rs} es la entrada (r, s) de $\mathcal{C}A$. Como la transformada coseno discreta inversa bidimensional de una matriz X es $\mathcal{C}^{-1}X = C^t X C$, entonces

$$\begin{aligned} A &= \mathcal{C}^{-1}(\mathcal{C}A) \\ &= C^t(\mathcal{C}A)\,C = C^t\left(\sum_{r,s} \lambda_{rs} E_{rs}\right) C = \sum_{r,s} \lambda_{rs} C^t E_{rs} C = \sum_{r,s} \lambda_{rs} \mathcal{C}^{-1} E_{rs}. \end{aligned}$$

Vemos que la imagen original (representada por medio de la matriz A) puede descomponerse como una combinación de las matrices $\mathcal{C}^{-1}E_{rs}$ y precisamente, las imágenes asociadas a estas matrices son las que aparecen en la figura 10. Además vemos que cuanto más abajo y más a la derecha están estas imágenes, son "menos uniformes". Veamos la razón de esta "menor uniformidad".

Vamos a denotar por $[A]_{uv}$ la entrada (u, v) de la matriz A. Como

$$\left[\mathcal{C}^{-1}E_{rs}\right]_{uv} = \sum_{k=0}^{n-1} [C^t]_{uk} [E_{rs}C]_{kv},$$

antes tenemos que ser capaces de calcular $[E_{rs}C]_{kv}$:

$$[E_{rs}C]_{kv} = \sum_{m=0}^{n-1} [E_{rs}]_{km} [C]_{mv} = \begin{cases} [C]_{sv} & \text{Si } r = k, \\ 0 & \text{si } r \neq k. \end{cases}$$

Luego

$$\left[\mathcal{C}^{-1}E_{rs}\right]_{uv} = \sum_{k=0}^{n-1} [C^t]_{uk} [E_{rs}C]_{kv} = [C^t]_{ur} [C]_{sv} = [C]_{ru} [C]_{sv}$$

Las entradas de la matriz C están escritas en (14). Y podemos ver que *cuanto mayores sean r y s, las frecuencias de $\mathcal{C}^{-1}E_{rs}$ son mayores.* Esta es la explicación de que en la figura 10, cuanto más cerca estemos de la posición "sureste", la uniformidad es menor.

Otra idea básica de la compresión JPEG es que a escalas pequeñas, los cambios deben ser pequeños. Por lo tanto, si tenemos un trozo pequeño de una imagen, entonces las altas frecuencias se pueden eliminar sin que perjudique la calidad de la imagen. Con esto, ya tenemos el esquema básico.

Imagen original → Dividir la imagen en trozos pequeños →
→ Eliminar las altas frecuencias de cada trozo pequeño → Juntar los trozos.

Normalmente, el sistema de compresión JPEG divide la imagen original en bloques de 8×8 píxeles (un tamaño realmente pequeño). El siguiente paso es "suavizar" cada bloque, es decir eliminar las altas frecuencias. ¿Cómo eliminamos las altas frecuencias de cada trozo 8×8?

Sea A una matriz 8×8 (que corresponde a un bloque cuadrado de 8 píxeles). Primero se calcula $\mathcal{C}A = CAC^t$. Como esta operación hay que hacerla para cada bloque, resulta que la matriz C solo hay que calcularla una sóla vez (de hecho ya está implementada y no hay que calcularla nunca, ya que viene "de fábrica"). Para cada entrada (r, s) de $\mathcal{C}A$ el sistema JPEG calcula $[\mathcal{C}A]_{rs}/q_{rs}$ y redondea el resultado al entero más próximo obteniendo una matriz R con "muchos ceros". Esta matriz R es la que se guarda. Como resulta que deseamos eliminar las frecuencias mayores, los números q_{rs} deben ser mayores a medida que r y s aumentan. La matriz $Q = (q_{rs})$ se suele llamar "cuantizador". Este es el paso donde se "pierde información"; pero información irrelevante. Un ejemplo de la elección de Q es

$$Q = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 40 & 57 & 69 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}.$$

Esta matriz se ha obtenido de forma empírica y ha sido usada con buenos resultados (pero pueden usarse otras matrices concretas). Véanse las las recomendaciones de la *International Telecommuncation Union* [3, anexo K, pág. 143]. Para representar la imagen comprimida, (es decir para recuperar la matriz original, aproximadamente) se multiplica R por Q elemento a elemento y este producto es el que se antitransforma.

Ejemplo 6. Consideremos el siguiente bloque:

$$\texttt{A} = \begin{bmatrix} 148 & 146 & 148 & 148 & 148 & 147 & 146 & 148 \\ 27 & 30 & 40 & 60 & 90 & 138 & 145 & 146 \\ 20 & 22 & 21 & 22 & 20 & 24 & 84 & 146 \\ 66 & 60 & 38 & 23 & 24 & 20 & 23 & 72 \\ 148 & 147 & 146 & 125 & 47 & 22 & 22 & 23 \\ 148 & 148 & 144 & 146 & 144 & 44 & 20 & 20 \\ 148 & 145 & 148 & 147 & 146 & 99 & 25 & 22 \\ 147 & 146 & 148 & 146 & 148 & 122 & 20 & 21 \end{bmatrix}.$$

Esta matriz corresponde a la imagen de la izquierda de la figura 11. A continuación calculamos la transformada discreta coseno bidimensional de `A` y la almacenamos en `B`. Si queremos formar la matriz cuya entrada (r, s) es B_{rs}/q_{rs}, y si previamente hemos guardado la matriz Q, se calcula mediante `B./Q` (no olvide que en Matlab u Octave, la división entrada a entrada se hace con ./ y no con /). Por último

redondeamos al entero más próximo obteniendo la matriz

$$\mathtt{R} = \mathtt{round}(\mathtt{B./Q}) = \begin{bmatrix} 45 & 11 & -3 & -1 & 1 & -1 & 0 & 0 \\ -3 & -21 & 8 & -1 & -1 & 1 & 0 & 0 \\ 14 & -3 & -4 & 2 & 0 & 0 & 0 & 0 \\ 12 & 7 & -2 & 0 & 0 & 0 & 0 & 0 \\ 2 & 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Es en este último paso de redondeo donde se ha perdido información. Pero observe que esta última matriz solo tiene 21 entradas no nulas. Así, en vez de almacenar 64 entradas númericas, solo tenemos que almacenar 21 entradas enteras. ¡Una reducción considerable!

Ahora tenemos que "deshacer el proceso" para reecuperar la imagen (o matriz) original. Eso sí, un poco distorsionada. Primero multiplicamos entrada a entrada `R` por `Q` y aplicamos la transformada inversa coseno discreta bidimensional a este último resultado. Por último redondeamos las entradas obteniendo

$$\begin{bmatrix} 137 & 135 & 139 & 144 & 147 & 155 & 157 & 146 \\ 44 & 46 & 54 & 65 & 83 & 112 & 141 & 156 \\ 2 & 8 & 8 & 9 & 21 & 50 & 91 & 128 \\ 68 & 73 & 58 & 35 & 20 & 12 & 29 & 65 \\ 140 & 149 & 130 & 100 & 72 & 25 & 2 & 25 \\ 148 & 164 & 152 & 141 & 127 & 66 & 15 & 26 \\ 138 & 153 & 142 & 147 & 155 & 92 & 25 & 27 \\ 145 & 154 & 135 & 143 & 160 & 95 & 18 & 15 \end{bmatrix}.$$

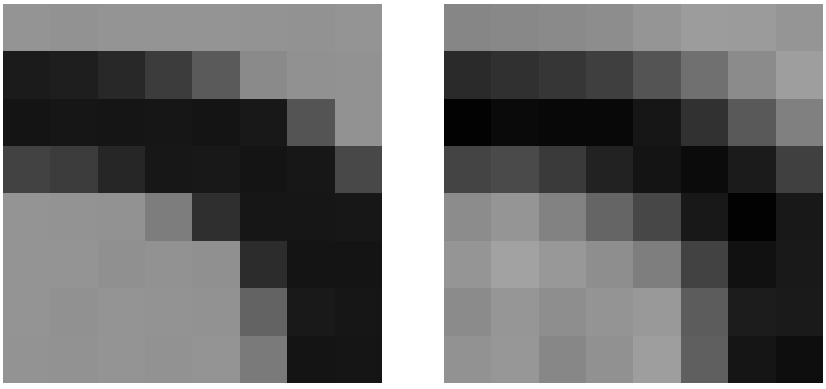

Figura 11: La imagen original y la imagen comprimida del ejemplo 6. Recuérdese que estos dos bloques son 8×8, y por tanto en una foto corriente son muy pequeños.

Agradecimiento

Quiero expresar mi más sincero agradecimiento al *referee* anónimo quien, con su extraordinario y completo informe, ha mejorado de manera notable la exposición, tanto matemática como de estilo, de este artículo.

Referencias

[1] Athanasios Papoulis. The Fourier Integral and its Applications. Mc-Graw-Hill.

[2] Carl D. Meyer. Matrix Analysis and Applied Linear Algebra. SIAM.

[3] `http://www.w3.org/Graphics/JPEG/itu-t81.pdf`

Julio Benítez López. Departamento de Matemática Aplicada. Instituto de Matemática Multidisciplinar. Universidad Politécnica de Valencia

Correo electrónico: `jbenitez@mat.upv.es`

Página web: `http://personales.upv.es/jbenitez`